```
REPORT  Z_MACROS1.

PARAMETER p_no type i.
DEFINE macro_print_natural_no.
 DO p_no TIMES.
  write / sy-index.
 ENDDO.
END-OF-DEFINITION.

macro_print_natural_no.
```

```
REPORT  Z_MACROS_WITH_ARGS1.

DEFINE FACT.
  &2 = 1.
  DO &1 TIMES.
    &2 = &2 * SY-INDEX.
    &1 = &1 + 1.
  ENDDO.
  END-OF-DEFINITION.

  FACT 5 1.
```

```
REPORT  Z_SUBROUTINES1.

PERFORM print_first_n_natural_numbers.
uline.

PARAMETER p_no type i.

form print_first_n_natural_numbers.
    DO p_no TIMES.
      write / sy-index.
    ENDDO.
endform.

* uline. * this statement is not accessible....
```

```
REPORT  Z_SUBROUTINES_USING_CHAINING2.

**** Call by value.....

DATA  v_counter TYPE I.
```

```
PERFORM increment_counter USING v_counter.
WRITE :/ 'The counter After increment is ', v_counter.

PERFORM increment_counter USING  v_counter.
WRITE :/ 'The counter After increment is ', v_counter.

PERFORM increment_counter USING  v_counter.
WRITE :/ 'The counter After increment is ', v_counter.

FORM increment_counter USING VALUE(fp_v_counter) TYPE I.

 fp_v_counter = fp_v_counter + 1.

WRITE :/ 'now fp_v_counter == ', fp_v_counter.
ENDFORM.
```

```
REPORT  Z_SUBROUTINES_USING_CHAINING3.
DATA v_counter TYPE i.

PERFORM increment_counter CHANGING v_counter.
WRITE :/ 'The counter After increment is ', v_counter.

PERFORM increment_counter CHANGING v_counter.
WRITE :/ 'The counter After increment is ', v_counter.

PERFORM increment_counter CHANGING v_counter.
WRITE :/ 'The counter After increment is ', v_counter.

FORM increment_counter CHANGING VALUE(fp_v_counter) TYPE I.

 fp_v_counter = fp_v_counter + 1.

write :/ 'now its ', fp_v_counter.
ENDFORM.
```

```
REPORT  Z_SUBROUTINES_WITH_ARGUEMENTS.

perform getfactorial USING 3.
PERFORM getfactorial USING 4.
PERFORM getfactorial USING 5.

FORM getfactorial USING x type I.
  DATA v_fact  TYPE I value 1.
```

```
  DO x TIMES.
   v_fact = v_fact * sy-index.
  ENDDO.
  write :/ ' factorial of ',x, '==========',  v_fact.
ENDFORM.
```

```
REPORT  Z_SUBROUTINES_WITH_TWO_ARGS.

data v1_fact type I.
data v2_fact type I.
data v3_fact type I.

perform getfactorial USING 3 v1_fact.
write :/ 'factorial of 3 is ', v1_fact.

PERFORM getfactorial USING 4 v2_fact.
write :/ 'factorial of 4 is ', v2_fact.

PERFORM getfactorial USING 5 v3_fact.
write :/ 'factorial of 5 is ', v3_fact.

FORM getfactorial USING x type I v_res type I.
  DATA v_fact  TYPE I value 1.

  DO x TIMES.
   v_fact = v_fact * sy-index.
  ENDDO.
 v_res = v_fact.
ENDFORM.
```